IEEE Control Systems Letters paper presented at
2023 American Control Conference (ACC)
San Diego, CA, USA. May 31 - June 2, 2023

Stabilization of a POD/Galerkin Reduced Order Payne-Whitham Traffic Model*

Brian Block, Xiaoling Chen, and Stephanie Stockar[1]

Abstract— This paper presents a method for generating stable reduced order macroscopic traffic models. Starting from the Payne-Whitham traffic model, a set of hyperbolic PDEs, Galerkin projection in conjunction with proper orthogonal decomposition is used. To enforce stability of the reduced order scheme, an extension of two methods developed for parabolic PDEs is presented in this paper. The performances of the a-posteriori stabilization schemes are compared showing that, with an appropriate calibration process, the stabilized models have comparable prediction errors to the full order models.

I. INTRODUCTION

The increasing number of vehicles on the road has resulted in longer travel times for drivers, increased energy consumption, and additional air pollution. In 2019 it was reported that in the United States, due to stop and go conditions, drivers spent an additional 8.7 billion hours on the road, consumed an additional 3.5 billion gallons of fuel, and emitted 36 million tons of excess greenhouse gases [1]. Controlling the inflow of vehicles by using macroscopic models has proven to be a successful strategy for reducing traffic jams [2]–[4].

Macroscopic models used to describe traffic flow are made up of partial differential equations (PDEs), where the dynamic variables are represented by locally aggregated quantities such as vehicle density, flow, and mean speed [5]. Common macroscopic models used for traffic simulation include first-order models such as the Lighthill-Whitham-Richards (LWR) model and second-order models such as the Payne-Whitham (PW) and Aw-Rascle-Zhang (ARZ) models [5]–[7]. The PW model is obtained by replacing the equilibrium velocity equation in the LWR model with a momentum equation [8]. There are drawbacks to the PW model such as negative speeds sometimes occuring at the tail-end of congested regions [9]. Second-order models are more realistic in capturing traffic dynamics in congestion due to the models possessing a family of flow rate curves [10].

For control and optimization applications, PDEs are usually converted into a set of coupled ordinary differential equations (ODEs) [11]–[14]. Model order reduction techniques used in conjunction with the ODE system include moment matching, balanced truncation, Galerkin projection, and proper orthogonal decomposition (POD). Compared to frequency-based methods, projection methods are best suited for traffic models as they retain accuracy over a wider range of speeds [15], [16]. For this paper, Galerkin projection with POD is considered due to the nonlinearities in the general traffic PDEs, and POD is of particular interest because real world data can be used to determine the basis [15]. A preliminary application of POD-Galerkin for computationally efficient traffic simulation is presented in [17]. However, no stability analysis of the reduced order model (ROM) was performed despite the fact that ROMs obtained by applying POD-Galerkin projection are not guaranteed to be stable even if the full order model (FOM) is stable [16], [18].

In literature, three approaches have been developed for enforcing stability of ROMs a-posteriori, of which the first two are conducive to control design. The first method utilizes pole placement to move the new and artificial unstable poles of the system to the stable region [18]. While conceptually simple, this method relies on the arbitrary selection of the pole location. Alternatively, a semidefinite programming problem can be solved, such that the reduced state matrix is asymptotically stable within a certain margin [16]. The computation time required for this method is higher compared to the pole-placement approach, but has the advantage of reducing the manual tuning. Finally, [18] also presents a method where the solution of a nonlinear least squares problem that minimizes the deviation between the ROM and FOM is used to reassign the unstable eigenvalues. This method provides a systematic approach to the stabilization of the ROM, however is computationally expensive as both the ROM and FOM need to be simulated simultaneously. This is a limitation for control design, as the process should be repeated every time a new controller is implemented, which changes the response of the system.

In this paper, a ROM of the linear PW model is obtained using POD-Galerkin and the stability of the resulting ROM is analyzed for different orders of truncation. To address the loss of stability caused by the model reduction process, two methods originally applied to parabolic PDEs, namely pole placement and semidefinite programming, are extended to hyperbolic problems in this paper and applied for the stabilization of the reduced PW model. A comparison between the different approaches is then presented.

II. HOMOGENEOUS PAYNE-WHITHAM MODEL

The homogeneous, nonlinear PW model is described by

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} = 0 \tag{1}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{c_0^2}{\rho}\frac{\partial \rho}{\partial x} = 0 \tag{2}$$

*This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1343012 and NSF CAREER Award 2042354 .

[1]The authors are with the Department of Mechanical and Aerospace Engineering and the Center for Automotive Research, The Ohio State University, 930 Kinnear Road, Columbus, OH 43212 USA block.168@osu.edu

979-8-3503-2806-6/$31.00 ©2023 AACC

where $\rho = \rho(x,t)$ is the traffic density, $u = u(x,t)$ is the velocity, and c_0 is maximum traffic wave speed. The linearized model is obtained by defining the perturbation variables:

$$\rho(x,t) = \rho_0 + \delta\rho(x,t) \tag{3}$$

$$u(x,t) = u_0 + \delta u(x,t) \tag{4}$$

where ρ_0 and u_0 are the equilibrium values for density and velocity, respectively, and $\delta\rho$ and δu are the perturbations around the equilibrium states. The linear PW model is

$$\frac{\partial\delta\rho}{\partial t} + \rho_0\frac{\partial\delta u}{\partial x} + u_0\frac{\partial\delta\rho}{\partial x} = 0 \tag{5}$$

$$\frac{\partial\delta u}{\partial t} + u_0\frac{\partial\delta u}{\partial x} + \frac{c_0^2}{\rho_0}\frac{\partial\delta\rho}{\partial x} = 0 \tag{6}$$

The two boundary conditions for the second-order PDE system are

$$\delta\rho(0,t) = \delta\rho_0(t) \tag{7}$$

$$\delta u(L,t) = K\delta\rho(L,t) \tag{8}$$

which consist of a Dirichlet boundary condition for density perturbation at the beginning of the road and an orifice boundary condition describing the relation between density and velocity perturbations at the end of the road, which is a common convention in gas dynamics [19]. These boundary conditions mimic the presence of infrastructure that restricts the flow of traffic, such as a traffic light, stop sign, or reduction in speed limit. The length of the road L and the parameter K are constant. For conciseness, since the rest of this section deals with the linear model, the perturbed variables, $\delta\rho$ and δu, will be given as ρ and u.

A. Analytical Solution of the Linear PW Model

The analytical solution of the linear PW model is found via a direct Laplace transform with zero initial conditions and is used to validate the FOM and ROM results. After the transform, the PW model in the frequency domain is

$$sP(x,s) + \rho_0\frac{dU(x,s)}{dx} + u_0\frac{dP(x,s)}{dx} = 0 \tag{9}$$

$$sU(x,s) + u_0\frac{dU(x,s)}{dx} + \frac{c_0^2}{\rho_0}\frac{dP(x,s)}{dx} = 0 \tag{10}$$

The two boundary conditions in the Laplace domain are

$$P(0,s) = P_0(s) \tag{11}$$

$$U(L,s) = KP(L,s) \tag{12}$$

Following the method found in [20], the transfer function between the perturbation in density at the end of length of road $P(L,s)$ and the beginning of length of road $P(0,s)$ is

$$\frac{P(L,s)}{P(0,s)} = \frac{z_2(c_0^2 - u_0^2)}{u_0(z_1 + K\rho_0 z_3) - \rho_0(\frac{c_0^2}{\rho_0}z_3 + Kz_1)} \tag{13}$$

where

$$z_1 = s(e^{m_2L} - e^{m_1L}) - u_0m_1e^{m_1L} + u_0m_2e^{m_2L} \tag{14}$$

$$z_2 = m_1e^{(m_1+m_2)L} - m_2e^{(m_1+m_2)L} \tag{15}$$

$$z_3 = -m_1e^{m_1L} + m_2e^{m_2L} \tag{16}$$

and

$$m_1 = \frac{s}{c_0 - u_0},\; m_2 = \frac{-s}{c_0 + u_0} \tag{17}$$

B. Numerical Solution of the Linear PW Model

The PW model defined in (5), (6) is semi-discretized in space using a first-order Euler finite difference scheme. This semi-discretization transforms the original PDEs into a system of ODEs, where the conservation of mass equation and momentum equation are staggered [19]. The resulting repeating structure of the ODEs at all interior nodes is

$$\frac{d\rho_i}{dt} = \frac{1}{\Delta x}[\rho_0(u_{i-1} - u_i) + u_0(\rho_{i-1} - \rho_i)] \tag{18}$$

$$\frac{du_{i-1}}{dt} = \frac{1}{\Delta x}[u_0(u_{i-1} - u_i) + \frac{c_0^2}{\rho_0}(\rho_{i-1} - \rho_i)] \tag{19}$$

where $i \in [2, n]$ since the density perturbation at the beginning of the road $\rho_1(t) = \rho(0,t)$ is the boundary condition. Using the boundary condition at the end of the length of road defined in (8), the remaining two ODEs for the density at n and velocity at $n-1$ are

$$\frac{d\rho_n}{dt} = \frac{1}{\Delta x}[\rho_0u_{n-1} + u_0\rho_{n-1} - (K\rho_0 + u_0)\rho_n] \tag{20}$$

$$\frac{du_{n-1}}{dt} = \frac{1}{\Delta x}[u_0u_{n-1} + \frac{c_0^2}{\rho_0}\rho_{n-1} - (Ku_0 + \frac{c_0^2}{\rho_0})\rho_n] \tag{21}$$

The velocity at the last node $u_n(t)$ is given by the orifice equation in (8). Using the perturbed densities ρ_i and velocities u_i at the same discretized spatial locations as the state variables, the system of ODEs can be written in state space:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{22}$$

The input to the system is

$$u = \rho_1 \tag{23}$$

The state vector is

$$x = \begin{bmatrix} \rho_2 & u_1 & \rho_3 & u_2 & \cdots & \rho_n & u_{n-1} \end{bmatrix}^T \tag{24}$$

The system $A \in R^{2(n-1)\times 2(n-1)}$, $B \in R^{2(n-1)\times 1}$, and $C \in R^{1\times 2(n-1)}$ is written explicitly as

$$\begin{aligned} A &= \frac{1}{\Delta x}\begin{bmatrix} -u_0 & \rho_0 & 0 & -\rho_0 & 0 & 0 & \cdots \\ -\frac{c_0^2}{\rho_0} & u_0 & 0 & -u_0 & 0 & 0 & \cdots \\ u_0 & 0 & -u_0 & \rho_0 & 0 & -\rho_0 & \cdots \\ \frac{c_0^2}{\rho_0} & 0 & -\frac{c_0^2}{\rho_0} & u_0 & 0 & -u_0 & \cdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & u_0 & 0 & -K\rho_0 - u_0 & \rho_0 \\ 0 & 0 & 0 & \frac{c_0^2}{\rho_0} & 0 & -Ku_0 - \frac{c_0^2}{\rho_0} & u_0 \end{bmatrix} \\ B &= \frac{1}{\Delta x}\begin{bmatrix} u_0 & \frac{c_0^2}{\rho_0} & 0 & \cdots & 0 \end{bmatrix}^T \\ C &= \frac{1}{\Delta x}\begin{bmatrix} 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \end{aligned} \tag{25}$$

The output of the system is the density at the end of the road. This matches the analytical solution used as the benchmark. A comparison between the system responses obtained from the analytical and numerical solution is performed in the

Laplace domain using the parameters summarized in Tab. I. The direct comparison between the analytical and numerical solutions is shown in Fig. 1. Results show that the numerical solution matches both the amplitude and peak locations of the analytical solution over the frequency range of interest.

TABLE I

MODEL PARAMETERS FOR VERIFICATION.

Parameter	Value	Unit
Average density ρ_0	120	[cars/km]
Average velocity u_0	10	[m/s]
Maximum speed c_0	11.11	[m/s]
Restriction coefficient $K = u_0/\rho_0$	83	[m^2/ (cars·s)]
Road length L	6000	[m]
Spatial discretization n	601	[-]

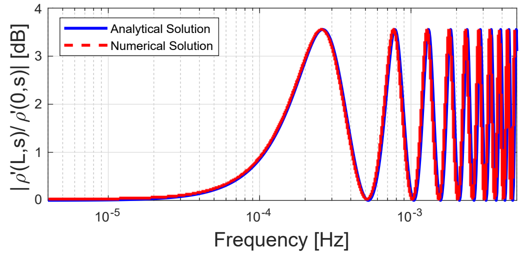

Fig. 1. System responses for the analytical and numerical solution.

C. Comparison between Linear and Nonlinear PW Models

To confirm the small perturbation assumption, the linear model described by Eq. (22)-(25) is compared against the nonlinear PW model defined in (1), (2) . The nonlinear model is solved using the MacCormack method [21], while the linear model is solved in MATLAB. The parameters for the simulation are summarized in Tab. I. The simulation is run for T = 600s with initial density ρ(x, 0) = 120 cars/km and velocity specified differently in two regions

$$u(x,0) = \begin{cases} 9, & \text{for } 0 \le x \le 2000 \\ 9 + 2 \cdot \tanh \frac{x-2000}{500\pi}, & \text{for } 2000 < x \le L \end{cases} \quad (26)$$

The boundary condition at the beginning of the road is ρ(0, t) = 120 cars/km. The point-to-point percentage error in density between the nonlinear model and the linear model is defined as

$$\text{error}(x,t) = \frac{\rho_{\text{nonlin}}(x,t) - \rho_{\text{lin}}(x,t)}{\max\limits_{x,t}(\rho_{\text{nonlin}}(x,t))} \cdot 100\% \quad (27)$$

The error is shown in Fig. 2, where minimal deviation between the linear and nonlinear models is observed. The maximum percentage error is -2.38%. The error starts to grow at around x = 2000m because the values of density there are getting further from the equilibrium value for density. Hence, a small perturbation assumption is appropriate for the selected case study.

III. REDUCED ORDER MODEL

A. Definition of the Case Study

The model parameters for the case study presented in this section are the same as those in Tab. I, except for the length of the road and spatial discretization. The new values are L = 500m and n = 51, respectively. The matrices A and B are the same as the ones defined in (25), with A ⍰ $R^{100\times100}$ and B ⍰ $R^{100\times1}$. Because both density and velocity are wanted at each discretized point along the road, C = I_{100} ⍰ $R^{100\times100}$. The initial conditions are

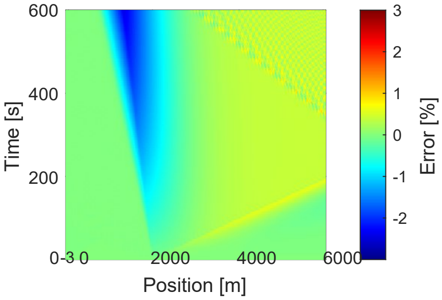

Fig. 2. Comparison of nonlinear and linear PW model.

$$\rho(x,0) = 0, \; u(x,0) = 0 \quad (28)$$

which corresponds to a completely empty road. This initial condition was chosen for consistency since the snapshot used later for the creation of the basis was formed starting from the zero initial condition. The boundary condition is

$$\rho(0,t) = \begin{cases} t/300, & \text{for } t \le 30 \\ 0.2 - t/300, & \text{for } t > 30 \end{cases} \quad (29)$$

which simulates an increase in traffic density up until t = 30s where it then decreases back. This simulates a traffic light upstream of x = 0m turning green at t = 0s and then turning red at t = 30s, restricting the flow of traffic. The resulting density profile for the nonlinear model is shown on the left in Fig. 3. As the traffic that enters the system after t = 30s starts to encounter the high density traffic already in the system, the traffic density starts to build up. This can be seen at around t = 50s.

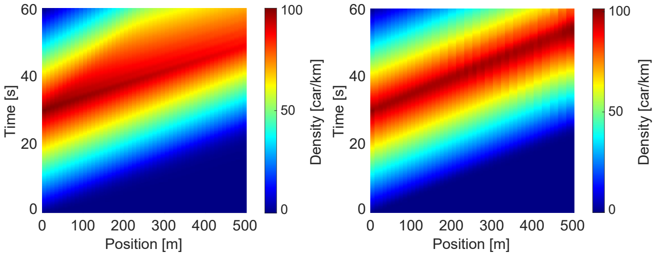

Fig. 3. (Left) Nonlinear result and (Right) ROM result, k = 30.

B. Petrov-Galerkin Projection

Assuming that the state trajectories x(t) can be sufficiently approximated by a trial basis V , it is possible to rewrite the state vector as

$$x(t) \approx V \cdot x_k(t) \quad (30)$$

The Petrov-Galerkin projection determines the basis V such that the residual r is constrained to be orthogonal to a subspace W defined by a test basis $W \in \mathbb{R}^{N\times k}$ [15]. For a LTI system, if V is nonsingular, then the matrices of the ROM are

$$A_k = W^T A V \quad (31)$$

$$B_k = W^T B \quad (32)$$

$$C_k = C V \quad (33)$$

where V, $W \in \mathbb{R}^{2(n-1)\times k}$ is an orthonormal vector, $A_k \in \mathbb{R}^{k\times k}$, $B_k \in \mathbb{R}^{k\times l}$, and $C_k \in \mathbb{R}^{q\times k}$ [15]. The dimensions l, q, and k represent the input size, output size, and the order of the ROM. A special case of the Petrov-Galerkin projection is the Galerkin projection, where $V = W$.

C. Proper Orthogonal Decomposition

The basis used for the projection is determined empirically through POD [15]. The snapshot matrix, S, is formed by simulating the linear model with the initial conditions given by (28) and a step function for the boundary condition:

$$\rho(0,t) = \begin{cases} 0, & \text{for } 0 \leq t < 10 \\ 0.066, & \text{for } 10 \leq t \leq 80 \end{cases} \quad (34)$$

The resulting values of density and velocity are used to populate S. Singular value decomposition (SVD) is then applied to the resulting snapshot matrix S. The left singular vectors of the matrix S are used to populate the matrices V and W.

D. Reduced Order Model Results

A reduced model with truncation order $k = 30$ is simulated and results are shown on the right side of Fig. 3. Compared to the result from the nonlinear model, the ROM is able to achieve the same profile. Where the models differ is the dissipation of the maximum density line. As shown in the nonlinear density profile, the high density region, in dark red, starts to dissipate as the traffic reaches $x = 300$m, whereas in the ROM result the high density region does not dissipate. Moreover, and as noted before, the ROM density profile does not show the build up of traffic density that is observed in the nonlinear result. The relative time it takes to simulate the ROM in comparison with the nonlinear model shown is Fig. 4, for each reduced order averaged over 20 runs. Both models are simulated using MATLAB on a 3.4GHz Intel Xeon CPU with 32 GB of RAM. As expected, the ROMs outperform the nonlinear model in terms of simulation time, with the ROMs being at least 20 times faster up to $k = 30$. To evaluate the accuracy of the ROM, the root mean square error (RMSE) of the simulated density is calculated:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n_x}\sum_{j=1}^{n_t}(\rho_{FOM} - \rho_{ROM})^2}{n_x \cdot n_t}} \quad (35)$$

where ρ_{FOM} is the density from the FOM, ρ_{ROM} is the density from the ROM, n_x is the number of spatial nodes, and n_t is the number of time steps. The RMSE for all reduced orders is shown in Fig. 5. As is expected, the error between the FOM and ROM decreases as the order of the ROM increases. For this case study, however, it is observed that as the order of the ROM increases, the model becomes unstable despite the original FOM being stable. This behaviour is seen in Fig. 5 for orders 36 to 67. Even though the RMSE for orders 36-40 is still low, the model is still unstable as the reduced order matrix A_k has positive eigenvalues. It is worth noting that the trend of unstable modes for higher orders of truncation is specific to this case study. The stability of the ROM is highly dependent on the choice of the equilibrium values ρ_0 and u_0 and the maximum speed c_0. To showcase this, 3 different values of u_0 are used, and the RMSE and stability of the ROMs is evaluated for the same initial conditions. The resulting RMSE for each case is shown in Fig. 6. For both $u_0 = 9$m/s and $u_0 = 10$m/s, the value used in this paper, the ROM goes unstable at $k = 37$, but for $u_0 = 11$m/s the model does not go unstable until $k = 44$. So, depending on the equilibrium values used in the linearization procedure, the stability of the the ROM changes. Moreover, the instability of the system can only be determined a-posteriori, hence after the reduction. So, a stabilization procedure that produces consistent, stable ROMs is necessary in order to use reduced order traffic models for control purposes. For the rest of the paper the values given in Tab. I will be used.

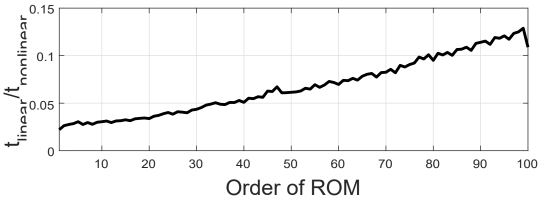

Fig. 4. Relative simulation time for the ROMs.

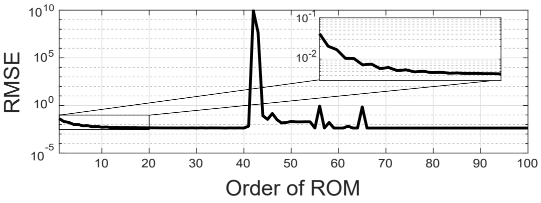

Fig. 5. RMSE for the Reduced Order Models.

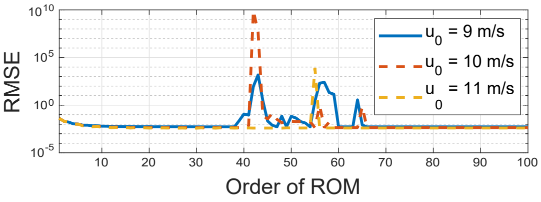

Fig. 6. Error for unstable ROMs.

IV. STABILIZATION PROCEDURE

A. Stabilization via Pole Placement

The method developed in [18] uses full state feedback to stabilize unstable ROMs. This is equivalent to introducing an artificial viscosity in the system. A control matrix, B_c, is chosen and applied to the ROM:

$$\begin{aligned} \dot{x}_k &= A_k x_k + B_k u + B_c u_c \\ y_k &= C_k x_k \end{aligned} \tag{36}$$

In this case, u_c is assumed to be a linear state feedback control input of the form

$$u_c = K_c x_k \tag{37}$$

Substituting (37) into (36) produces

$$\begin{aligned} \dot{x}_k &= (A_k - B_c K_c) x_k + B_k u \\ y_k &= C_k x_k \end{aligned} \tag{38}$$

where the closed-loop system matrix is

$$\tilde{A}_k = A_k - B_c K_c \tag{39}$$

In this case study, B_c is chosen to be the identity matrix, I_k, such that the controllability matrix for the pair (A_k, B_c) is full rank. The feedback matrix K_c is computed such that $\tilde{A}_k$ has stable poles. The desired poles of the closed-loop system are found by reassigning the eigenvalues:

$$\begin{aligned} \lambda_k^s &\Rightarrow \lambda_k^s \\ \lambda_k^u &\Rightarrow -\alpha \cdot \mathrm{Re}(\lambda_k^u) \pm i \cdot \mathrm{Im}(\lambda_k^u) \end{aligned} \tag{40}$$

where λ_k^s are the stable eigenvalues, λ_k^u are the the unstable eigenvalues, and $\alpha > 0$ is a modifiable parameter. For this case study, α was chosen to be 0.1 1 10 .

B. Stabilization via Semidefinite Programming

An alternative to pole placement is presented in [16], where the ROM is minimally modified to ensure stability. The reduced order bases (ROBs) in this method are created from a truncation of the larger computed set from POD. The procedure starts from a left ROB W_{k+p} and searches for a left ROB $\tilde{W}_k \in \mathrm{range}(W_{k+p})$ where p is the additional search size and k is the order of the ROM, as before. $\tilde{W}_k$ can be written as

$$\tilde{W}_k = W_{k+p} X \tag{41}$$

where $X \in \mathbb{R}^{(k+p)\times k}$ is a matrix with full column rank. The stabilization procedure takes an unstable ROM of the form

$$\begin{aligned} E_k \dot{x} &= A_k x_k + B_k u \\ y_k &= C_k x_k \end{aligned} \tag{42}$$

where $E_k = I_k$, and converts it to

$$\begin{aligned} \dot{x} &= \tilde{A}_k x_k + \tilde{B}_k u \\ y_k &= \tilde{C}_k x_k \end{aligned} \tag{43}$$

where

$$\tilde{A}_k = \tilde{E}_k^{-1} X^T (M_A - \mu M_E) \tag{44}$$

$$\tilde{B}_k = \tilde{E}_k^{-1} X^T B_{k+p} \tag{45}$$

$$\tilde{C}_k = C_{k+p} \tag{46}$$

$$\tilde{E}_k = X^T M_E \tag{47}$$

The matrices $M_E \in \mathbb{R}^{(k+p)\times k}$ and $M_A \in \mathbb{R}^{(k+p)\times k}$ are the reduction of the matrices $E = I$ and A, respectively, using W_{k+p}, and are given by

$$M_E = W_{k+p}^T E V_k \tag{48}$$

$$M_A = W_{k+p}^T (A + \mu E) V_k \tag{49}$$

The parameter μ is a stability margin which is an upper bound placed on the real parts of the eigenvalues of the system. The matrix X is found by solving an optimization problem using semidefinite programming (SDP). The optimization problem is given as

$$\begin{aligned} \min_{\hat{P} \in \mathbb{R}^{(k+p)\times(k+p)}} \quad & f(\hat{P}) \\ \text{s.t.} \quad & M_E^T \hat{P} M_A + M_A^T \hat{P} M_E = -Q \\ & \hat{P} > 0_{(k+p)\times(k+p)} \end{aligned} \tag{50}$$

It is shown in [16] that if $\hat{P} \in \mathbb{R}^{(k+p)\times(k+p)}$ exists then a positive semidefinite matrix $\tilde{P} \in \mathbb{R}^{(k+p)\times(k+p)}$ of rank k can be constructed such that

$$M_E^T \tilde{P} M_A + M_A^T \tilde{P} M_E = -Q \tag{51}$$

where $\tilde{P}$ is given as

$$\tilde{P} = X \hat{P}_{11}^{-1} X^T \tag{52}$$

The matrices $\hat{P}$ and $\tilde{P}$ can be written as

$$\hat{P} = \begin{matrix} \hat{P}_{11} & \hat{P}_{12} \\ \hat{P}_{12}^T & \hat{P}_{22} \end{matrix} \tag{53}$$

$$\tilde{P} = \begin{matrix} \hat{P}_{11} \\ \hat{P}_{12}^T \end{matrix} \hat{P}_{11}^{-1} \begin{matrix} \hat{P}_{11} & \hat{P}_{12} \end{matrix} \tag{54}$$

It follows then, from (52) and (54), that

$$X = \begin{matrix} \hat{P}_{11} \\ \hat{P}_{12}^T \end{matrix} \tag{55}$$

The objective function f can be written as

$$f = \hat{P}_{12}^T \quad \hat{P}_{22}\,{}_F + \tau \hat{P}_{11}\,{}_F \tag{56}$$

where $\tau \hat{P}_{11}$ is a regularization term that is added so that $\hat{P}_{11}$ does not grow without bounds. The optimization problem is solved in MATLAB using CVX [22]. For the case study, the values of the parameters were chosen to be $\mu = 0$, $Q = I_k$, and $\tau = 10^{-5}$. At higher ROM orders, the value for p had to be increased so that the algorithm was able to find a stable solution. For $k \in [37, 40]$, using $p = 5$ led to a stabilized result, but for orders greater than $k = 40$, the minimization problem was unfeasible and $p = 10$ was used.

C. Comparison

The comparison of the two methods, together with the unstabilized ROMs, is shown in Fig. 7. For the stable ROMs, all stabilization methods show the same result as they do not change the dynamics of the stable ROMs.

For pole placement, the value of the RMSE varies greatly with the selected value of α. For the unstable models between orders $k = 37$ and $k = 40$, where originally the error was low in spite of the instability, α = 0.1 and α = 1 are enough for keeping the error low while still stabilizing the model. However, choosing α = 10 leads to a higher error in spite of the stabilization. Another example of the variation in performance is at k = 44 where α = 10 results in the lowest error, but at k = 45 the same selection results in the highest error. This study shows that there is no systematic approach for selecting the value of α that provides consistent, low error over a span of ROM orders.

SDP, on the other hand, gives more consistent results, even if the value chosen for p is not optimal. One drawback with SDP is that as the model becomes larger, a higher value of p must be chosen in order to find a solution. The larger the value of p, the more time it takes to stabilize using SDP. With pole placement, since it is just reassigning eigenvalues, the process of stabilization is not computationally intensive for any value of k. In light of control applications, both methods suffer due to the fact that they stabilize the ROM a-posteriori. As well, since the case study was based on a linear model, it is unknown how these methods would work on the nonlinear PW model.

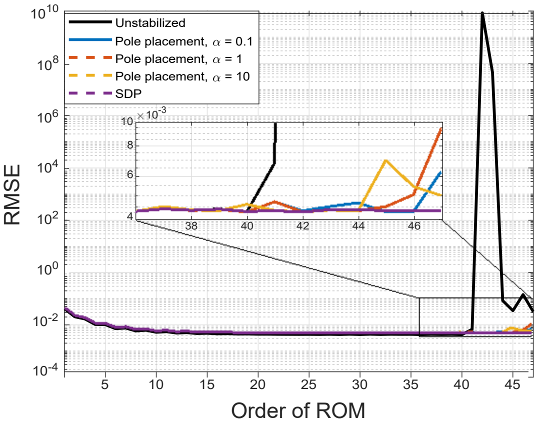

Fig. 7. Comparison of RMSE for all methods.

V. CONCLUSION

This paper presented a method for generating stable reduced order models of the Payne-Whitham traffic model. First, the homogeneous PW model was linearized and the numerical solution was compared against the analytical solution. Moreover, a comparison between linear and nonlinear versions was shown for a selected case study. Starting from the linear model, a reduced model was obtained via POD/Galerkin projection. Results show that, in some cases, even for a stable, full order model, the ROM can become unstable as a result of the projection process. To remedy this, two stabilization methods were extended to hyperbolic equations, one that utilizes pole placement and one that utilizes semidefinite programming. The methods were compared, and it was shown that the method based on semidefinite programming provided better overall stabilization for the POD/Galerkin reduced PW model. However, both methods are limited as they stabilize the ROMs a-posteriori which is undesirable for control applications. The results were shown on a linear model, but an extension to a more realistic nonlinear traffic model, such as the ARZ model, is needed to be able to better describe traffic and to handle shocks, in the form of traffic jams, in congested traffic.

REFERENCES

[1] D. Schrank, T. Lomax, and B. Eisele, “2021 urban mobility report,” Texas Transportation Institute, 2021. [Online]. Available: http://mobility.tamu.edu/ums/report

[2] H. Yu and M. Krstic, “Traffic congestion control for Aw-Rascle-Zhang model,” Automatica, vol. 100, pp. 38–51, 2019.

[3] H. Yu, S. Park, A. Bayen, S. Moura, and M. Krstic, “Reinforcement learning versus PDE backstepping and PI control for congested freeway traffic,” arXiv preprint arXiv:1904.12957, 2019.

[4] H. Yu, M. Diagne, L. Zhang, and M. Krstic, “Bilateral boundary control of moving shockwave in LWR model of congested traffic,” IEEE Transactions on Automatic Control, vol. 66, no. 3, pp. 1429–1436, 2020.

[5] M. Treiber and A. Kesting, Traffic Flow Dynamics: Data, Models and Simulation. Springer Science & Business Media, 2012.

[6] V. Knoop, Introduction to Traffic Flow Theory: An introduction with exercises. TU Delft Open, 2017.

[7] A. Aw and M. Rascle, “Resurrection of second order models of traffic flow,” SIAM Journal on Applied Mathematics, vol. 60, no. 3, pp. 916–938, 2000.

[8] W. Jin and H. M. Zhang, “Solving the Payne-Whitham traffic flow model as a hyperbolic system of conservation laws with relaxation,” Transportation Science, 2001.

[9] C. F. Daganzo, “Requiem for second-order fluid approximations of traffic flow,” Transportation Research Part B: Methodological, vol. 29, no. 4, pp. 277–286, 1995.

[10] S. Fan, M. Herty, and B. Seibold, “Comparative model accuracy of a data-fitted generalized aw-rascle-zhang model,” Networks & Heterogeneous Media, vol. 9, no. 2, p. 239, 2014.

[11] P. D. Christofides and P. Daoutidis, “Robust control of hyperbolic PDE systems,” Chemical engineering science, vol. 53, no. 1, pp. 85–105, 1998.

[12] J. A. Atwell, J. T. Borggaard, and B. B. King, “Reduced order controllers for burgers’ equation with a nonlinear observer,” International Journal of Applied Mathematics and Computer Science, vol. 11, pp. 1311–1330, 2001.

[13] N. H. El-Farra, A. Armaou, and P. D. Christofides, “Analysis and control of parabolic PDE systems with input constraints,” Automatica, vol. 39, no. 4, pp. 715–725, 2003.

[14] E. W. Sachs and S. Volkwein, “POD-Galerkin approximations in PDE-constrained optimization,” GAMM-Mitteilungen, vol. 33, no. 2, pp. 194–208, 2010.

[15] A. C. Antoulas, Approximation of large-scale dynamical systems. SIAM, 2005.

[16] D. Amsallem and C. Farhat, “Stabilization of projection-based reduced-order models,” International Journal for Numerical Methods in Engineering, vol. 91, no. 4, pp. 358–377, 2012.

[17] Z. Luo, D. Xie, and F. Teng, “A POD-based reduced-order FD extrapolating algorithm for traffic flow,” Advances in Difference Equations, vol. 2014, no. 1, pp. 1–13, 2014.

[18] I. Kalashnikova, B. van Bloemen Waanders, S. Arunajatesan, and M. Barone, “Stabilization of projection-based reduced order models for linear time-invariant systems via optimization-based eigenvalue reassignment,” Computer Methods in Applied Mechanics and Engineering, vol. 272, pp. 251–270, 2014.

[19] S. Stockar, M. Canova, Y. Guezennec, A. Della Torre, G. Montenegro, and A. Onorati, “Model-order reduction for wave propagation dynamics in internal combustion engine air path systems,” International Journal of Engine Research, vol. 16, no. 4, pp. 547–564, 2015.

[20] E. Doebelin, System Dynamics: Modeling, Analysis, Simulation, Design. CRC Press, 1998.

[21] J. D. Anderson, Computational Fluid Dynamics: The Basics with Applications. McGraw-Hill, 1995.

[22] M. Grant and S. Boyd, “CVX: Matlab software for disciplined convex programming, version 2.1,” 2014. [Online]. Available: http://cvxr.com/cvx